Consolidating Streams to Improve DASH Cache Utilisation

Stephen McQuistin
School of Computing Science
University of Glasgow
sm@smcquistin.uk

Colin Perkins
School of Computing Science
University of Glasgow
csp@csperkins.org

ABSTRACT

Existing HTTP caches interact poorly with multiple Dynamic Adaptive Streaming over HTTP (DASH) streams of the same content: time and quality differences prevent a complete representation from being cached, reducing hit-ratios. We propose to consolidate near-simultaneous streams based on time or quality, where the improved cache performance makes this worthwhile. We estimate that there is a sufficient number of near-simultaneous streams for our proposed techniques to improve cache hit-ratios.

CCS Concepts

•Networks → Application layer protocols; Middle boxes / network appliances;

Keywords

DASH; cache; hit-ratio

1. INTRODUCTION

The use of HTTP-based adaptive streaming protocols, including DASH and HTTP Live Streaming (HLS), is increasing. The compatibility of these protocols with the existing HTTP infrastructure, including caches, drives this growth. Adaptive streaming protocols offer multiple encodings of the same content, split into chunks, with receivers selecting the most appropriate chunk to request at a given time.

These algorithms do not adopt a holistic view of video delivery: they are designed to request the best quality chunk at any given time, without regard for other users, or caches in the network. This behaviour leads to reduced cache hit-ratios [4], and a lower quality of experience for users of the cache.

CoNEXT Student Workshop '15, December 01-01 2015, Heidelberg, Germany

ACM 978-1-4503-4066-3/15/12.

http://dx.doi.org/10.1145/2842665.2843567

In this paper, we propose to consolidate DASH streams based on two factors: time, and bitrate representation. Multiple clients that are requesting the same content at negligible time offsets or at different bitrates may have their flows combined to increase cache hit-ratios.

The stream consolidation techniques that we propose make use of standard HTTP redirections, without any endpoint changes, increasing deployability. Other work in this area increases cache utilisation by prefetching chunks that clients are likely to request in future; the techniques proposed here are complementary to these strategies.

Section 2 outlines the proposed techniques. Section 3 presents analysis of when the techniques may be beneficial. Section 4 describes related work, and Section 5 concludes.

2. STREAM CONSOLIDATION

The strategies outlined in this section require that the cache be able to retrieve and interpret the media presentation description (MPD) for the content being requested. This describes URL structure for the content, and the representations offered, allowing the cache to issue the correct HTTP redirects. Lee et al. [6] describe methods for MPD retrieval.

Broadly, the techniques we propose serve clients (using a standard HTTP redirect) with an alternative chunk, either of a different bitrate representation, or time in the stream.

The need to inspect and modify end-to-end DASH flows does not necessarily conflict with the ongoing shift towards mass encryption on the Internet. Content providers can adopt encryption mechanisms that allow for encryption to be used alongside the techniques we propose [3] [8].

2.1 Time-based consolidation

As a client streams media using DASH, the cache will store a moving window of chunks, with the most recently requested chunk at the right edge. The size of this window depends on caching policies, but will be limited by the capacity of the cache. If another client streams the same content, then it will be served from the cache only if the requested chunks fall within the window of an existing flow. Otherwise, a new moving window will be created within the cache, reducing the sizes of all other windows, and negating the benefit of the cache for that client.

We propose to consolidate streams of the same content

(at the same bitrate representation), but at time offsets that are sufficient to negate the cache, yet are not significant for the user. Identifying streams that should be consolidated is a function of the time offset between flows, content duration, remaining duration, number of streams, and cache capacity. Other metrics may be useful in identifying users that may be sensitive to the any disruption this technique introduces. For example, Krishnan et al. [5] suggest that users streaming at higher bitrate representations are more sensitive to disruptions.

Implementation of this technique requires careful consideration to ensure that the cache utilisation benefits outweigh the disruption to the stream. A consideration when consolidating streams is the resultant user behaviour. If a user's stream skips ahead, then the user is missing content, and may attempt to seek to the point in the stream that they missed. However, if a user's stream jumps to a previous chunk, then they are seeing duplicate content, and should be less likely to intervene. Therefore, the consolidation algorithm should consolidate flows around played-out chunks. The use of this technique should be limited within a given stream.

2.2 Rate-based consolidation

Gouta et al. [4] show that offering multiple representations decreases cache hit-ratios by 15%, versus a single representation. This arises because the cache treats each representation separately, and chunks of one representation cannot serve requests for another representation. A complete representation is not available in the cache [7].

We propose to consolidate multiple streams that are requesting the same content simultaneously (i.e., not subject to the time-based consolidation technique above) at different representations. Not only does this allow for a greater number of chunks to be cached, but frees up upstream bandwidth, giving greater capacity to prefetching algorithms.

As with time-based consolidation, care needs to taken to ensure that user experience is maintained. Determining when to consolidate multiple streams based on quality is a function of the represention offered, the difference between the requested and consolidated representations, cache capacity, and the bandwidth capacity between the users, cache and content server.

3. PRELIMINARY ANALYSIS

The proposed techniques only improve performance when there are multiple requests for the same content, at different representations or small time offsets. We assume that there will be sufficient variation in the timing and representations of requests; here, we show that there is a non-trivial number of requests for the same content.

For this analysis, we will use data from the BBC iPlayer platform in the UK [1]. In July 2015, there was an average of 6.3 million requests for TV content per day. 483,000 (7%) of these requests were made at the peak time, with an average of 28,890 (0.6%) of these requests for the most requested content on the platform.

Assume a three level cache hierarchy, between sender and receiver, with caches placed at core points of presence (PoPs), access PoPs, and local exchanges, as shown in Figure 1. Using UK population density estimates, we estimate that each exchange cache will see around five near-simultaneous streams. Based on this, our proposed techniques might be thought to have limited utility. However, as we ascend the cache hierarchy, the benefit becomes clear: core caches see thousands of simultaneous streams.

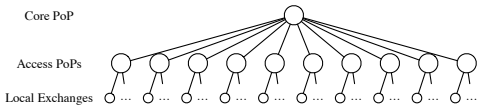

Figure 1: Cache hierarchy

The benefits of our proposal will only increase: current iPlayer usage is dwarfed by traditional TV consumption. As the number of streams delivered over IP grows to match that delivered using broadcast, caching becomes increasingly important to protect the network. The techniques we propose will remain useful for as long as live or scheduled TV is delivered by content providers.

It remains to show that the user experience impact of our techniques do not outweigh the benefits they deliver. The user experience of streaming video applications can be viewed as a trade-off between three measures: start-up time, frequency of rebuffering, and representation bitrate. The improved hit-ratios delivered by our techniques may allow caches to compensate for this trade-off. User studies are needed to show that the user experience impact is worthwhile.

4. RELATED WORK

CF-DASH [2] aims to reduce switches between multiple representations by allowing the cache and clients to determine a suitable bitrate. This bitrate is not enforced: clients may request a representation that negatively impacts the performance of the cache. This reduces deployability.

ViSIC [6] addresses the *bitrate oscillation* problem using traffic shaping techniques. The techniques used, such as the bandwidth estimation algorithms, may be useful for implementing the techniques proposed here.

Rejaie et al. [9] propose caching techniques based on content popularity, where popular content is afforded greater cache capacity. When the video is provided using a layered encoding, more layers can be cached, improving quality for the most popular content. This is not the case with DASH delivery, where different representations do not share layers.

5. CONCLUSION

We have proposed techniques designed to improve cache hit-ratios. It remains for these techniques to be implemented and evaluated, including investigating interactions with other caching policies, such as prefetching algorithms. Beyond showing that the goals with respect to cache utilisation are met, user studies are needed to demonstrate that any disruption is minimal and worthwhile.

6. REFERENCES

[1] iPlayer - Monthly Performance Pack, June and July 2015. BBC iStats, July 2015.

[2] Z. Aouini, M. T. Diallo, A. Gouta, A.-M. Kermarrec, and Y. Lelouedec. Improving Caching Efficiency and Quality of Experience with CF-Dash. In *Proceedings of Network and Operating System Support on Digital Audio and Video Workshop*, NOSSDAV '14, pages 61:61–61:66, Singapore, 2014. ACM.

[3] I. Brown, C. Perkins, and J. Crowcroft. Watercasting: Distributed watermarking of multicast media. In *Networked Group Communication*, pages 286–300. Springer, 1999.

[4] A. Gouta, D. Hong, A.-M. Kermarrec, and Y. Lelouedec. HTTP Adaptive Streaming in Mobile Networks: Characteristics and Caching Opportunities. In *Modeling, Analysis Simulation of Computer and Telecommunication Systems (MASCOTS), 2013 IEEE 21st International Symposium on*, pages 90–100, Aug 2013.

[5] S. S. Krishnan and R. K. Sitaraman. Video stream quality impacts viewer behavior: inferring causality using quasi-experimental designs. *Networking, IEEE/ACM Transactions on*, 21(6):2001–2014, 2013.

[6] D. H. Lee, C. Dovrolis, and A. C. Begen. Caching in HTTP adaptive streaming: Friend or foe? In *Proceedings of the International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*, Singapore, March 2014. ACM.

[7] C. Liu, M. M. Hannuksela, and M. Gabbouj. Client-driven joint cache management and rate adaptation for Dynamic Adaptive Streaming over HTTP. *International Journal of Digital Multimedia Broadcasting*, 2013.

[8] D. Naylor, K. Schomp, M. Varvello, I. Leontiadis, J. Blackburn, D. R. López, K. Papagiannaki, P. Rodriguez Rodriguez, and P. Steenkiste. Multi-Context TLS (mcTLS): Enabling Secure In-Network Functionality in TLS. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, SIGCOMM '15, pages 199–212, London, United Kingdom, 2015. ACM.

[9] R. Rejaie, H. Yu, M. Handley, and D. Estrin. Multimedia proxy caching mechanism for quality adaptive streaming applications in the internet. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 980–989. IEEE, 2000.